Intel® Celeron® M Processor

Specification Update

June 2004

Notice: The Intel® Celeron® M processor may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are documented in this Specification Update.

Document Number: 300303-004

Contents

Revision History

Revision Number	Description	Date
001	Initial Release	January 2004
002	Revisions include: • Added errata WW22- W23 • Added Specification Clarification W1 • Updated Processor Identification table	April 2004
003	Revisions include: • Added errata WW24- W25	May 2004
004	Revisions include: • Updated Processor Identification table	June 2004

Preface

This document is an update to the specifications contained in the following documents:

- *Intel® Celeron® M Processor datasheet(order number 300302)*
- *Intel® Architecture Software Developer's Manual, Volumes 1, 2, and 3* (Order Numbers 243190, 243191, and 243192, respectively)

It is intended for hardware system manufacturers and software developers of applications, operating systems, or tools. It contains S-Specs, Errata, Documentation Changes, Specification Clarifications and Specification Changes.

Nomenclature

S-Spec Number is a five-digit code used to identify products. Products are differentiated by their unique characteristics, e.g., core speed, L2 cache size, package type, etc. as described in the processor identification information table. Care should be taken to read all notes associated with each S-Spec number.

Errata are design defects or errors. Errata may cause the Intel® Celeron® M processor's behavior to deviate from published specifications. Hardware and software, designed to be used with any given processor, must assume that all errata documented for that processor are present on all devices unless otherwise noted.

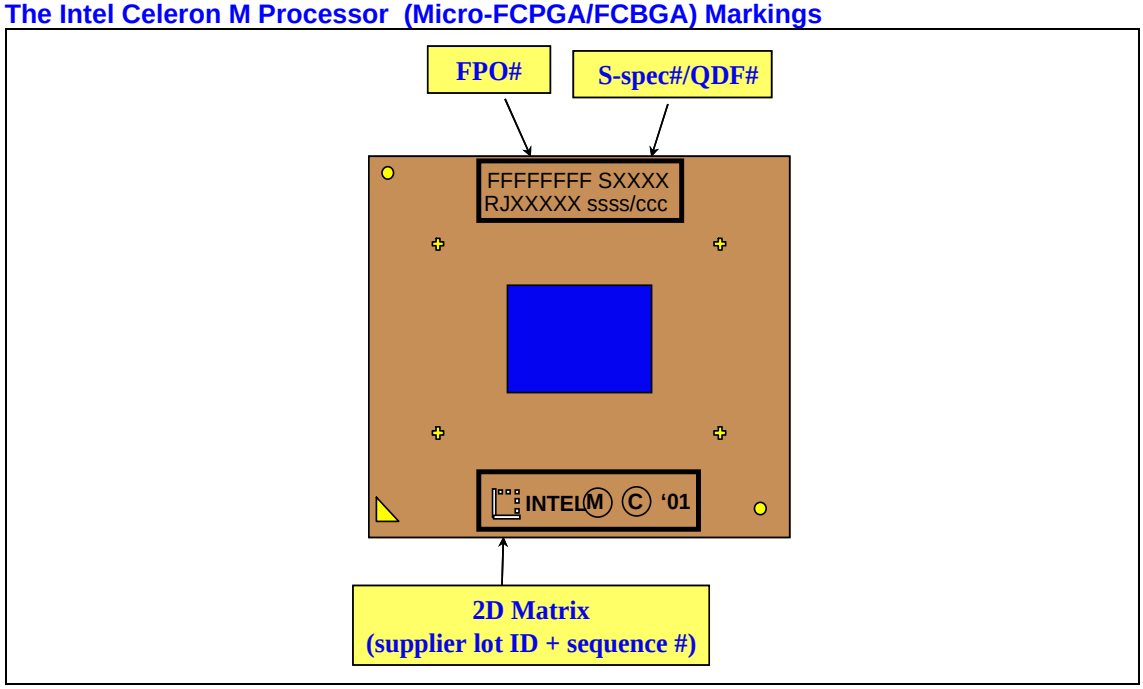
Documentation Changes include typos, errors, or omissions from the current published specifications. These changes will be incorporated in the next release of the specifications.

Specification Clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in the next release of the specifications.

Specification Changes are modifications to the current published specifications for the Intel® Celeron® M processor. These changes will be incorporated in the next release of the specifications.

General Information

Figure 1. The Intel Celeron M Processor (Micro-FCPGA/FCBGA) Markings



Identification Information

The Intel Celeron M processor can be identified by the following values:

Family[1]	Model[2]	Brand ID[3]
0110	1001	00010010

NOTES:

1. The Family corresponds to bits [11:8] of the EDX register after Reset, bits [11:8] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
2. The Model corresponds to bits [7:4] of the EDX register after Reset, bits [7:4] of the EAX register after the CPUID instruction is executed with a 1 in the EAX register.
3. The Brand ID corresponds to bits [7:0] of the EBX register after the CPUID instruction is executed with a1 in the EAX register.

Table 1. Intel Celeron M Processor Identification

S-Spec/ QDF	Product Stepping	L2 Cache Size (Kbytes)	CPUID	Core Frequency	Bus Frequency	Voltage	Package	Notes
SL6N7	B-1	512K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCPGA	
SL6NM	B-1	512K	0695h	1.30 GHz	400 MHz	1.356 V	Micro-FCBGA	
SL6N6	B-1	512K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCPGA	
SL6NL	B-1	512K	0695h	1.40 GHz	400 MHz	1.356 V	Micro-FCBGA	
SL7MT	B-1	512K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCBGA	
SL7ME	B-1	512K	0695h	1.50 GHz	400 MHz	1.356 V	Micro-FCPGA	
SL79S	B-1	512K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCPGA	
SL79T	B-1	512K	0695h	1.20 GHz	400 MHz	1.356 V	Micro-FCBGA	
SL7GE	B-1	512K	0695h	600 MHz	400 MHz	1.004 V	Micro-FCBGA	1
SL7DB	B-1	512K	0695h	800 MHz	400 MHz	1.004 V	Micro-FCBGA	
SL7DH	B-1	512K	0695h	900 MHz	400 MHz	1.004 V	Micro-FCBGA	

NOTES:
1. This product is for customers of the Embedded Intel Architecture Division

Summary Tables of Changes

The following table indicates the Errata, Documentation Changes, Specification Clarifications, or Specification Changes that apply to Intel Celeron M processors. Intel intends to fix some of the errata in a future stepping of the component and to account for the other outstanding issues through documentation or specification changes as noted. This table uses the following notations:

Codes Used in Summary Table

Stepping

X:	Erratum, Specification Change or Clarification that applies to this stepping.
(No mark) or (Blank Box):	This erratum is fixed in listed stepping or specification change does not apply to listed stepping.

Status

Doc:	Document change or update that will be implemented.
PlanFix:	This erratum may be fixed in a future of the product.
Fixed:	This erratum has been previously fixed.
NoFix:	There are no plans to fix this erratum.
Shaded:	This item is either new or modified from the previous version of the document.

Each Specification Update item is prefixed with a capital letter to distinguish the product. The key below details the letters that are used in Intel's microprocessor Specification Updates:

A = Intel® Pentium® II processor
B = Mobile Intel® Pentium® II processor
C = Intel®Celeron® processor
D = Intel® Pentium® II Xeon™ processor
E = Intel® Pentium® III processor
G = Intel® Pentium® III Xeon™ processor
H = Mobile Intel® Celeron® processor at 466/433/400/366/333/300 and 266 MHz
K = Mobile Intel® Pentium® III Processor-M
M = Mobile Intel® Celeron® processor
N = Intel® Pentium® 4 processor
O = Intel® Xeon™ processor MP
P = Intel® Xeon™ processor
T = Mobile Intel® Pentium® 4 Processor-M
V = Mobile Intel® Celeron® processor on .13 Micron Process in Micro-FCPGA Package
W = Intel® Celeron® M processor
Y = Intel® Pentium® M processor
Z = Mobile Intel® Pentium® 4 processor with 533 MHz system bus

The Specification Updates for the Pentium® processor, Pentium® Pro processor, and other Intel products do not use this convention.

NO.	Steppings	Plans	ERRATA
	B1		
W1	X	NoFix	Performance Monitoring Event that counts the number of instructions decoded (D0h) is not accurate
W2	X	NoFix	RDTSC Instruction May Report the Wrong Time Stamp Counter Value
W3	X	NoFix	Code Segment limit violation may occur on 4 Gbyte limit check
W4	X	NoFix	FST Instruction with Numeric and Null Segment Exceptions may cause General Protection Faults to be Missed and FP Linear Address (FLA) Mismatch
W5	X	NoFix	Code Segment is wrong on SMM Handler when SMBASE is not aligned
W6	X	NoFix	IFU/BSU Deadlock May Cause System Hang
W7	X	NoFix	Processor can enter a livelock condition under certain conditions when FP exception is pending.
W8	X	NoFix	Write Cycle of Write Combining Memory Type does not Self Snoop
W9	X	NoFix	Performance Monitoring Event that counts Floating Point Computational Exceptions (11h) is not accurate.
W10	X	NoFix	Inconsistent Reporting of Data Breakpoints on FP (MMX) loads
W12	X	NoFix	Code Breakpoint may be taken after POP SS instruction if it is followed by an instruction that faults
W12	X	NoFix	SysEnter and SysExit instructions may write incorrect Requestor Privilege Level (RPL) in the FP Code Segment selector (FCS)
W13	X	NoFix	Memory Aliasing with Inconsistent A and D Bits may Cause Processor Deadlock
W14	X	NoFix	RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault
W15	X	NoFix	FP Tag Word Corruption
W16	X	NoFix	Unable to Disable Reads/Writes to Performance Monitoring Related MSRs
W17	X	NoFix	Move to Control Register Instruction May Generate a Breakpoint Report
W18	X	NoFix	REP MOVS Operation in Fast String Mode Continues in That Mode When Crossing Into a Page With a Different Memory Type
W19	X	NoFix	The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page With a UC Memory Type
W20	X	NoFix	Machine Check Exception May Occur Due to Improper Line Eviction in the IFU
W21	X	NoFix	POPF and POPFD Instructions That Set the Trap Flag Bit May Cause Unpredictable Processor Behavior
W22	X	NoFix	Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event
W23	X	NoFix	VM Bit Will Be Cleared on a Double Fault Handler
W24	X	NoFix	Code Fetch Matching Disabled Debug Register May Cause Debug Exception
W25	X	NoFix	Upper Four PAT Entries Not Usable With Mode B or Mode C Paging

SPEC CHANGE NUMBER	STATUS	SPECIFICATION CHANGE
		There are no specification changes in this Specification Update revision.

SPEC CHANGE NUMBER	STATUS	SPECIFICATION CLARIFICATIONS
		There are no specification clarificatios in this Specification Update revision

NO.	DOCUMENT REVISION	STATUS	DOCUMENTATION CHANGES
			There are no documentation changes in this Specification Update revision.

This page intentionally left blank.

Errata

W1. Performance Monitoring Event that Counts the Number of Instructions Decoded (D0h) is Not Accurate

Problem: The performance-monitoring event that counts the number of instructions decoded may have inaccurate results.

Implication: There is no functional impact of this erratum. However, the results/counts from this Performance Monitoring Event should not be considered as being accurate

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W2. RDTSC Instruction May Report the Wrong Time Stamp Counter Value

Problem: The Time Stamp Counter is a 64-bit counter that is read in two 32-bit chunks. The counter incorrectly advances and therefore the two chunks may go out of synchronization causing the Read Time Stamp Counter (RDTSC) instruction to report the wrong time stamp counter value

Implication: This erratum may cause software to see the wrong representation of processor time and may result in unpredictable software operation.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W3. Code Segment Limit Violation May Occur on 4-Gbyte Limit Check

Problem: Code Segment limit violation may occur on 4-Gbyte limit check when the code stream wraps around in a way that one instruction ends at the last byte of the segment and the next instruction begins at 0x0.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Avoid code that wraps around segment limit.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W4. FST Instruction with Numeric and Null Segment Exceptions May Cause General Protection Faults to be Missed and FP Linear Address (FLA) Mismatch

Problem: FST instruction combined with numeric and null segment exceptions may cause General Protection Faults to be missed and FP Linear Address (FLA) mismatch.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W5. Code Segment is Wrong on SMM Handler When SMBASE is Not Aligned

Problem: With SMBASE being relocated to a non-aligned address, during SMM entry the CS can be improperly updated which can lead to an incorrect SMM handler.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: Align SMBASE to 32 Kbyte.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W6. IFU/BSU Deadlock May Cause System Hang

Problem: A lockable instruction with memory operand that spans across two pages may, given some rare internal conditions, hang the system.

Implication: When this erratum occurs, the system may hang. Intel has not observed this erratum with any commercially available software or system.

Workaround: Lockable data should always be contained in a single page.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W7. Processor Can Enter a Livelock Condition Under Certain Conditions When FP Exception is Pending

Problem: Processor clock modulation may be controlled via a processor register (IA32_THERM_CONTROL) or via the STPCLK# signal. While the processor clock is constantly being actively modulated at 12.5% and 25% duty cycles and there is a pending unmasked FP exception (ES pending), if you attempt a FP load (or MMX Mov instruction) and the load has an longer than typical latency the processor can enter a livelock.

Implication: When this erratum occurs, the processor will enter a livelock condition. Intel has not observed this erratum with any commercially available software or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W8. Write Cycle of Write Combining Memory Type Does Not Self Snoop

Problem: Write cycles of WC memory type does not self-snoop. This may result in data inconsistency- if the addresses of the WC data are aliased to WB memory type memory, which has been cached. In such a case, the internal caches will not be updated with the WC data sent on the system bus.

Implication: This condition may result in a data inconsistency. Intel has not observed this erratum with any commercially available software, system, nor components.

Workaround: Software should detect via the self-snoop bit in the CPUID features flags if the processor supports a self-snooping capability. Software should perform explicit memory management/flushing for aliased memory ranges on processors that do not self-snoop.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W9. Performance Monitoring Event that counts Floating Point Computational Exceptions (11h) is Not Accurate

Problem: Performance monitoring event that counts Floating Point Compare exceptions may have inaccurate results.

Implication: There is no functional impact of this erratum. However this Performance Monitoring Event should not be used when accurate performance monitoring is required.

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W10. Inconsistent Reporting of Data Breakpoints on FP (MMX) Loads

Problem: The reporting of data breakpoints on either FP or MMX loads is dependent upon the code faulting behavior prior to the execution of the load. If there is a fault pending prior to the execution of the load and FP exceptions are enabled there is a chance that data breakpoint on successive FP/MMX Loads may be reported twice.

Implication: Software debuggers should be aware of this possibility. There should be no implications to software operated outside of a debug environment.

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W11. Code Breakpoint May Be Taken After POP SS Instruction if it is Followed by an Instruction That Faults

Problem: A POP SS instruction should inhibit all interrupts including Code Breakpoints until after execution of the following instruction. This allows sequential execution of POP SS and MOV eSP, eBP instructions without having an invalid stack during interrupt handling. However, a Code breakpoint may be taken after POP SS if it is followed by an instruction that faults, this results in a code breakpoint being reported on an unexpected instruction boundary since both instructions should be atomic.

Implication: This can result in a mismatched Stack Segment and SP. Intel has not observed this erratum with any commercially available software, or system.

Workaround: As recommended in the *IA32 Intel® Architecture Software Developer's Manual*, the use "POP SS" in conjunction with "MOV eSP, eBP" will avoid the failure since the "Mov" will not fault.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W12. SysEnter and SysExit Instructions May Write Incorrect Requestor Privilege Level (RPL) in the FP Code Segment Selector (FCS)

Problem: SysEnter and SysExit instructions may write incorrect RPL in the FP Code Segment selector (FCS). As a result of this, the RPL field in FCS may be corrupted.

Implication: This is a rare condition that may result in a system hang. Intel has not observed this erratum with any commercially available software, or system.

Workaround: It is possible for BIOS to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W13. Memory Aliasing with Inconsistent A and D Bits May Cause Processor Deadlock

Problem: In the event that software implements memory aliasing by having two page directory entries (PDEs) point to a common page table entry (PTE) and the Accessed and Dirty bits for the two PDEs are allowed to become inconsistent the processor may become deadlocked.

Implication: This erratum has not been observed with commercially available software.

Workaround: Software that needs to implement memory aliasing in this way should manage the consistency of the Accessed and Dirty bits.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W14. RDMSR or WRMSR to Invalid MSR Address May Not Cause GP Fault

Problem: The RDMSR and WRMSR instructions allow reading or writing of MSR's(Model Specific Registers) based on the index number placed in ECX. The processor should reject access to any reserved or unimplemented MSRs by generating #GP(0). However, there are some invalid MSR addressers for which the processor will not generate #GP(0). This erratum has not been observed with commercially available software.

Implication: For RDMSR, undefined values will be read into EDX:EAX. For WRMSR, undefined processor behavior may result.

Workaround: Do not use invalid MSR addresses with RDMSR or WRMSR.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W15. FP Tag Word Corruption

Problem: In some rare cases, fault information generated as the result of instruction execution may be incorrect. The result is an incorrect FP stack entry.

Implication: This erratum may result in corruption of the FP Tag Word in a way that a non-valid entry in the FP Stack may become valid. The software is not expected to read a non-valid entry. If the software attempts to use the stack entry (which is expected to be empty) the result may be an erroneous "Stack overflow".

Workaround: Do not disable SSE/SSE2 in control register CR4 and avoid code segment limit violation.

Status: For the steppings affected, see the *Summary of Table of Changes*.

W16. Unable to Disable Reads/Writes to Performance Monitoring Related MSRs

Problem: The Performance Monitoring Available bit in the miscellaneous processor features MSR (IA32_MISC_ENABLES.7) was defined so that when it is cleared to a 0, RDMSR/WRMSR/RDPMC instructions would return all zeros for reads of and prevent any writes to performance monitoring related MSRs. Currently it is possible to read from or write to performance monitoring related MSRs when the Performance Monitoring Available bit is cleared to a 0.

Implication: It is not possible to disallow reads and writes to the Performance Monitoring MSRs. Intel has not observed this erratum with any commercially available software or system.

Workaround: None.

Status: For the steppings affected, see the *Summary of Table of Changes.*

W17. Move to Control Register Instruction May Generate a Breakpoint Report

Problem: A move (MOV) to Control Register (CR) instruction where Control Register is CR0, CR3 or CR4 may generate a breakpoint report.

Implication: MOV to Control Register Instruction is not expected to generate a breakpoint report.

Workaround: Ignore breakpoint data from MOV to CR instruction.

Status: For the steppings affected, see the *Summary of Table of Changes.*

W18. REP MOVS Operation in Fast String Mode Continues in That Mode When Crossing Into a Page With a Different Memory Type

Problem: A fast “REP MOVS” operation will continue to be handled in fast mode when the string operation crosses a page boundary into an Uncacheable (UC) memory type. Also if the fast string operation crosses a page boundary into a WC memory region, the processor will not self-snoop the WC memory region. This may eventually result in incorrect data for the WC portion of the operation if those cache lines were previously cached as WB (through aliasing) and modified.

Implication: String elements should be handled by the processor at the native operand size in UC memory. In the event that the WB to WC aliasing case occurs, the end result could vary from normal software execution to potential software failure. Intel has not observed either aspects of this erratum in commercially available software.

Workaround: Software operating within Intel’s recommendation will not require WB and WC memory aliased to the same physical address.

Status: For the steppings affected, see the *Summary Table of Changes.*

W19. The FXSAVE, STOS, or MOVS Instruction May Cause a Store Ordering Violation When Data Crosses a Page With a UC Memory Type

Problem: If the data from an FXSAVE, STOS, or MOVS instruction crosses a page boundary from WB to UC memory type and this instruction is immediately followed by a second instruction that also issues a store to memory, the final data stores from both instructions may occur in the wrong order.

Implication: The impact of this store ordering behavior may vary from normal software execution to potential software failure. Intel has not observed this erratum in commercially available software.

Workaround: FXSAVE, STOS, or MOVS data must not cross page boundary from WB to UC memory type.

Status: For the steppings affected, see the *Summary Table of Changes.*

W20. Machine Check Exception May Occur Due to Improper Line Eviction in the IFU

Problem: The processor is designed to signal an unrecoverable Machine Check Exception (MCE) as a consistency checking mechanism. Under a complex set of circumstances involving multiple speculative branches and memory accesses there exists a one cycle long window in which the processor may signal a MCE in the Instruction Fetch Unit (IFU) because instructions previously decoded have been evicted from the IFU. The one cycle long window is opened when an opportunistic fetch receives a partial hit on a previously executed but not as yet completed store resident in the store buffer. The resulting partial hit erroneously causes the eviction of a line from the IFU at a time when the processor is expecting the line to still be present. If the MCE for this particular IFU event is disabled, execution will continue normally.

Implication: While this erratum may occur on a system with any number of processors, the probability of occurrence increases with the number of processors. If this erratum does occur, a machine check exception will result. Note systems that implement an operating system that does not enable the Machine Check Architecture will be completely unaffected by this erratum (e.g., Windows* 95 and Windows 98).

Workaround: It is possible for BIOS code to contain a workaround for this erratum.

Status: For the steppings affected, see the *Summary of Table of Changes.*

W21. POPF and POPFD Instructions That Set the Trap Flag Bit May Cause Unpredictable Processor Behavior

Problem: In some rare cases, POPF and POPFD instructions that set the Trap Flag (TF) bit in the EFLAGS register (causing the processor to enter Single-Step mode) may cause unpredictable processor behavior.

Implication: Single step operation is typically enabled during software debug activities, not during normal system operation.

Workaround: There is no workaround for single step operation in commercially available software. For debug activities on custom software the POPF and POPFD instructions could be immediately followed by a NOP instruction to facilitate correct execution.

Status: For the steppings affected, see the *Summary of Table of Changes.*

* Other names and brands may be claimed as the property of others.

W22. Performance Event Counter Returns Incorrect Value on L2_LINES_IN Event

Problem: The performance event counter returns an incorrect value on L2_LINES_IN event (EMON event #24H) when the L2 cache is disabled.

Implication: Due to this erratum, L2_LINES_IN performance event counter should not be monitored while the L2 cache is disabled. This erratum has no functional impact.

Workaround: Ignore L2_LINES_IN event when the L2 cache is disabled.

Status: For the steppings affected, see the *Summary of Table of Changes.*

W23. VM Bit Will Be Cleared on a Double Fault Handler

Problem: Following a task switch to a Double Fault Handler that was initiated while the processor was in virtual-8086 (VM86) mode, the VM bit will be incorrectly cleared in EFLAGS.

Implication: When the OS recovers from the double fault handler, the processor will no longer be in VM86 mode

Workaround: None

Status: For the steppings affected, see the *Summary of Table of Changes.*

W24. Code Fetch Matching Disabled Debug Register May Cause Debug Exception

Problem: The bits L0-3 and G0-3 enable breakpoints local to a task and global to all tasks, respectively. If one of these bits is set, a breakpoint is enabled, corresponding to the addresses in the debug registers DR0-DR3. If at least one of these breakpoints is enabled, any of these registers are *disabled* (i.e., L*n* and G*n* are 0), and RW*n* for the disabled register is 00 (indicating a breakpoint on instruction execution), normally an instruction fetch will not cause an instruction-breakpoint fault based on a match with the address in the disabled register(s). However, if the address in a disabled register matches the address of a code fetch which also results in a page fault, an instruction-breakpoint fault will occur.

Implication: While debugging software, extraneous instruction-breakpoint faults may be encountered if breakpoint registers are not cleared when they are disabled. Debug software which does not implement a code breakpoint handler will fail, if this occurs. If a handler is present, the fault will be serviced. Mixing data and code may exacerbate this problem by allowing disabled data breakpoint registers to break on an instruction fetch.

Workaround: The debug handler should clear breakpoint registers before they become disabled.

Status: For the steppings affected, see the *Summary of Table of Changes.*

W25. Upper Four PAT Entries Not Usable With Mode B or Mode C Paging

Problem: The Page Attribute Table (PAT) contains eight entries, which must all be initialized and considered when setting up memory types for the Pentium III processor. However, in Mode B or Mode C paging, the upper four entries do not function correctly for 4-Kbyte pages. Specifically, bit 7 of page table entries that translate addresses to 4-Kbyte pages should be used as the upper bit of a 3-bit index to determine the PAT entry that specifies the memory type for the page. When Mode B (CR4.PSE = 1) and/or Mode C (CR4.PAE) are enabled, the processor forces this bit to zero when determining the memory type regardless of the value in the page table entry. The upper four entries of the PAT function correctly for 2-Mbyte and 4-Mbyte large pages (specified by bit 12 of the page directory entry for those translations).

Implication: Only the lower four PAT entries are useful for 4-KB translations when Mode B or C paging is used. In Mode A paging (4-Kbyte pages only), all eight entries may be used. All eight entries may be used for large pages in Mode B or C paging.

Workaround: None identified.

Status: For the steppings affected, see the *Summary of Table of Changes*

Specification Changes

There are no specification changes in this specification update revision.

Specification Clarifications

There are no specification clarifications in this specification update revision.

Documentation Changes

There are no documentation changes in this Specification Update revision.

§